DECUS

PROGRAM LIBRARY

DECUS NO.	11-398
TITLE	LSI-11 Audio Cassette Utility
AUTHOR	George Woods
COMPANY	Digital Equipment Corporation
DATE	June 1979
SOURCE LANGUAGE	MACRO-11

ATTENTION

This is a USER program. Other than requiring that it conform to submittal and review standards, no quality control has been imposed upon this program by DECUS.

The DECUS Program Library is a clearing house only; it does not generate or test programs. No warranty, express or implied, is made by the contributor, Digital Equipment Computer Users Society or Digital Equipment Corporation as to the accuracy or functioning of the program or related material, and no responsibility is assumed by these parties in connection therewith.

LSI-11 Audio Cassette Utility

11-398

Version: June 1979

Author: George Woods
Digital Equipment Corporation, Maynard, MA

Operating System: Stand-alone

Source Language: MACRO-11

Memory Required: 512 words RAM or ROM

Special Hardware Required: Any cassette recorder/player (or equivalent device) interfaced via a bus interface (user-supplied).

The Audio Cassette Utility (ACU) program is designed for use on LSI-11 systems for loading and saving memory images of programs and data on audio cassette tapes. The recorder/player is interfaced to the system via a simple interface circuit that connects to a DLV11 Serial Line Unit. Other media (e.g. paper tape) and device interface hardware can be used with this program as long as device registers are compatible with the DLV11.

Three ACU commands are provided for reading (loading), writing (dumping), and verifying (non-memory modifying read) program tapes in absolute loader format. In response to operator commands, ACU displays parameters that the user can change or insert before actually executing the requested operation.

One additional command effects a continuous series of alternate 1 and 0 bits to be output as an aid in tuning or testing audio cassette interface hardware.

All ACU code is PIC allowing the program to be relocated, starting on any word address. All ACU code can be loaded in either RAM or ROM; only 512 locations are required. ACU automatically establishes a four-word stack at the top of read-write memory, regardless of system size or location of the ACU program. Two start/restart addresses are provided, allowing ACU to be loaded in 512X4 PROMs as used on the MRV11-AA PROM module. Detailed loading and operating instructions are included in the ACU User's Guide documentation.

Also being distributed is an LSI-11 Audio Cassette Interface (only a write-up (A2) is available). This interface is designed to connect the DLV11 20 ma I/O to the audio cassette recorder EAR and AUX (or MIC) jacks.

Note: The binary tape supplied contains position-independent code and can be loaded at any starting address in RAM or ROM.

TABLE OF CONTENTS

ACU User's Guide 1
Abstract, ACU Command Summary 6
Definitions and Loading 7
Initialize 8
Prompt 9
Command Monitor Routine 10
Tune Routine 11
Read (Loader) Routine 12
Print Subroutine 14
Write Routine 15
Verify Routine 18
Message Buffers 20

AUDIO CASSETTE UTILITY (ACU) USER'S GUIDE

George Woods

July 2, 1979

Digital Equipment Corporation

INTRODUCTION

ACU was written to meet the need for an easy-to-use means for saving and restoring memory images and data in a home computer environment where expensive peripheral devices are not available. The actual computer hardware currently using this version of ACU is an LSI-11. A simple audio cassette interface converts DLV11 20 mA current loop I/O signals to frequency-modulated audio signals to/from the cassette recorder/player. The system is capable of operating at serial transfer rates up to 4800 bps, depending upon recorder and tape quality. Using high-quality tapes and typical recorders in the $20-$30 range, serial transfer rates of 2400 bps should easily be attained. No modifications to the recorder are required, although it may be desirable to modify the speaker circuit so that it remains connected when a cable is plugged into the EAR jack, allowing the operator to monitor audio tones.

In order to reduce the possibility of having bad recordings after the actual memory image is lost, a verify operation can be performed by ACU. This is a non memory-modifying read of the tape data in which each recorded byte is compared with the current memory contents.

All I/O transfers are in Absolute Loader format. Only one (large or small) data block is output with the usual checksum byte appended. The first two bytes on any record following the leader (null) bytes are byte values of 1 followed by 0; no other format is accepted by ACU.

LOADING AND STARTING ACU

ACU is primarily designed for use in PROM. However, ACU can be loaded into RAM using the Absolute Loader (paper tape) or via the AL or AR commands issued to the REV11-A or -C system options. When loading ACU into RAM, it is recommended that you relocate the program to the upper 512 words in RAM; load and start ACU at the relocation bias/starting address indicated:

Memory Size	Relocation bias/ Starting address
4K	16000
8K	36000
12K	56000
16K	76000
20K	116000
24K	136000
28K	156000
30K	166000

When loaded into PROM, ACU serves as a useful program to enter upon system power-up. In order to do this, the first instruction executed must be in location 173000 of the I/O page (Bank 7). Two entry points are provided in ACU: program addresses 0 and 1000. Normally, only the first address is used. However, if the MRV11-AA hardware is used, the second address enables the PROM-resident ACU program to be located on the MRV11-AA option configured for PROM addresses 172000-173776. Thus, upon power-up, the mid address in the PROM (172000+1000=173000) becomes accessed. This location contains a JMP instruction that returns program control to the normal program start/restart address.

ACU COMMAND SUMMARY

All ACU commands are entered following the ACU prompt (and Bell code) display. For example, the Read command (R key followed by the RETURN key) is entered as shown below:

```
ACU>R
```

ACU will typically display a message that prompts the operator to enter specific command parameters into CPU registers. Default values are generally included to reduce required operator intervention. The location of default values within ACU are listed later in this guide so that you may customize ACU for your specific system configuration; if you plan to change default values, be sure to insert the changes before loading PROMs.

ACU commands are listed below. Each command is described in detail in paragraphs that follow:

Command	Description
R	Read - Read (load) Absolute Loader format tapes and relocate (optional) as indicated
W	Write - Write (dump) Absolute Loader format tape of the specified memory area and append an autostart address (optional)
V	Verify - Verify a tape just written via a non-destructive read operation, comparing each byte to actual memory contents
T	Tune - Output a continuous stream of alternate 1 and 0 bits for maintenance purposes

Read Command

ACU responds to the Read command by displaying the following message and the CPU halts:

```
* ENTER LOAD BIAS IN R1 (DFLT=0)
* ENTER READER CSR IN R5 (DFLT=177560)
* START TAPE, PRESS P DURING LEADER
```

If relocated loading is not desired (normal procedure), do not enter a value in R1. If the default device CSR is correct for the actual reader being used, simply start the tape and press the P key to continue program execution; otherwise, enter the desired CSR address in R5 before continuing. A good load is indicated when the program autostarts or the ACU prompt is displayed. Checksum errors are reported by the *ERROR* message display. If an error occurs, reposition the tape to the beginning of the block and press the P key.

Write Command

ACU responds to the Write command by displaying the following message and the CPU halts:

```
* ENTER VALUES:
 R1=FIRST ADDR
 R2=LAST ADDR IN DUMP
 R3=AUTOSTART ADDR (DFLT=NO AUTOSTART)
 R4=DUMP DEVICE CSR (DFLT=177564)
 R5=LENGTH OF LEADER
* START TAPE, PRESS P
```

At this point, you must enter values in R1 and R2 to indicate the first and last addresses in the memory area to be saved. These are required parameters.

You can specify an autostart address by entering a valid starting address for the program you are saving in R3. Otherwise, the default value of 1 will indicate no autostart.

The dump device CSR can be specified by entering its address in R4. The default device is the console terminal.

The number of blank leader frames desired is indicated by the octal number entered in R5. The default value is 1700 (960 decimal) which produces two seconds of leader when running tape I/O at 4800 bps.

Once the desired parameter values have been entered, start the recorder and press the P key. When the write operation has been completed, the ACU prompt will be displayed. At this point you should verify the tape data before memory contents become lost or modified.

Verify Command

ACU responds to the Verify command by displaying the following message and the CPU halts:

```
        * ENTER READER CSR IN R5 (DFLT=177560)
        * START TAPE, PRESS P DURING LEADER
```

Position the previously written tape to the beginning of the tape (leader portion). If required, change the read device CSR parameter in R5. Start the tape and press the P key. ACU checks each data byte against the actual memory contents. A good tape is indicated by the display of the ACU prompt message. An error is indicated by the *ERROR* message display and the CPU halts. You can analyze the error, if desired, by examining the contents of CPU registers. The address of the current memory byte in error is contained in R0; the good data byte (actual memory data) is contained in R1; the bad byte read from tape is contained in R2. Return to the Write routine to produce a new tape by pressing the P key.

CHANGING DEFAULT VALUES

Default values can be changed in the ACU program in RAM prior to saving it on cassette tape or punching paper tape masters for PROM hardware. Operation on your system can be simplified by changing the default values to meet your specific requirements. Default values are located at the addresses shown which are relative to the start of the program:

```
                        .TITLE  LSI-11/AUDIO CASSETTE UTILITY -- ACU X02
                        .SBTTL ABSTRACT, ACU COMMAND SUMMARY
;
;WRITTEN IN JUNE 1979 BY GEORGE R. WOODS OF DIGITAL EQUIPMENT CORPORATION
;MAYNARD, MASSACHUSETTS.

;
; **********************************************************************
;
; THIS  PROGRAM IS DESIGNED TO BE PROM OR RAM RESIDENT FOR USE IN LSI-11
; SYSTEMS WITH SIMPLE CASSETTE INTERFACE HARDWARE.  ALL  PROGRAM  LOAD,
; DUMP,  AND  VERIFY  OPERATIONS  ARE  IN  ABSOLUTE  LOADER FORMAT.  ALL
; PROGRAM CONTENT IS POSITION INDEPENDENT.   A  STACK  IS  AUTOMATICALLY
; ESTABLISHED AT THE TOP OF READ/WRITE MEMORY.  THUS, THE AUDIO CASSETTE
; UTILITY (ACU) PROGRAM CAN RESIDE IN ANY PORTION OF  READ/WRITE  MEMORY
; OR  READ-ONLY  MEMORY  (INCLUDING READ-ONLY MEMORY RESIDENT IN THE I/O
; PAGE).
;
; **********************************************************************
;
;
;     ****************************************************
;     *                                                  *
;     *       ACU COMMAND SUMMARY                        *
;     *                                                  *
;     *       R <CR>  READ COMMAND - READ ABSOLUTE       *
;     *               LOADER FORMAT TAPES FROM USER-     *
;     *               SPECIFIED DEVICE                   *
;     *                                                  *
;     *       W <CR>  WRITE COMMAND - OUTPUT AN ABSOLUTE *
;     *               LOADER FORMAT IMAGE OF A MEMORY    *
;     *               AREA TO A USER-SPECIFIED DEVICE,   *
;     *               WITH OR WITHOUT AUTOSTART          *
;     *                                                  *
;     *       V <CR>  VERIFY COMMAND - NON MEMORY MODIFY *
;     *               READ FROM USER-SPECIFIED DEVICE TO *
;     *               CHECK TAPE DATA CONFORMITY WITH    *
;     *               MEMORY CONTENTS                    *
;     *                                                  *
;     *       T <CR>  TUNE COMMAND - OUTPUT A SERIES OF  *
;     *               ALTERNATING 1 AND 0 BITS TO USER-  *
;     *               SPECIFIED DEVICE WHEN TUNING AUDIO *
;     *               CASSETTE INTERFACE HARDWARE        *
;     *                                                  *
;     ****************************************************
;
```

```
                                                    .SBTTL  DEFINITIONS AND LOADING
                                    ;
                                    ;       *****************************************************
                                    ;       *                                                   *
                                    ;       *       DEFINITIONS:                                *
                                    ;       *                                                   *
                                    ;       *****************************************************
                                    ;
                                    ;
            177560                          RCSR=177560
            177562                          RBUF=177562
            177564                          XCSR=177564
            177566                          XBUF=177566
                                    ;
                                    ;
                                    ;       *****************************************************
                                    ;       *                                                   *
                                    ;       *       LOADING AND STARTING ACU IN RAM:            *
                                    ;       *                                                   *
                                    ;       *       (LOAD AND START AT ADDRESS SHOWN)           *
                                    ;       *                                                   *
                                    ;       *       MEMORY                 LOAD/START           *
                                    ;       *       AVAILABLE              ADDRESS              *
                                    ;       *                                                   *
                                    ;       *           4K                  16000               *
                                    ;       *           8K                  36000               *
                                    ;       *          12K                  56000               *
                                    ;       *          16K                  76000               *
                                    ;       *          20K                 116000               *
                                    ;       *          24K                 136000               *
                                    ;       *          28K                 156000               *
                                    ;       *          30K                 166000               *
                                    ;       *                                                   *
                                    ;       *****************************************************
                                    ;
                                    ;
                                    ;
```

```
  1                                             .SBTTL  INITIALIZE
  2                                     ;
  3                                     ;       ****************************************************
  4                                     ;       *                                                  *
  5                                     ;       *       INITIALIZE:                                *
  6                                     ;       *                                                  *
  7                                     ;       ****************************************************
  8                                     ;
  9                                     ;
 10                                     ;
 11 000000  000005                      START:  RESET                   ;INITIALIZE BUS
 12                                     ;
 13                                             ;REGISTERS USED/MODIFIED:
 14                                             ;
 15                                             ;       R0 = TEMP SAVED LOCATION 4 CONTENTS
 16                                             ;       R1 = TEMP SAVED LOCATION 6 CONTENTS
 17                                             ;       R4 = TEMP POINTER TO TIMEOUT VECTOR
 18                                             ;       R5 = TRIAL ADDRESS POINTER
 19                                             ;       R6 = DUMMY DESTINATION/FINAL SP
 20                                     ;
 21 000002  012705  170000                      MOV     #170000,R5      ;MAXIMUM LSI-11 (MSV11-D/E) READ/WRITE ADDRESS+2
 22 000006  013700  000004                      MOV     @#4,R0          ;SAVE CONTENTS OF LOCATION 4
 23 000012  013701  000006                      MOV     @#6,R1          ;SAVE CONTENTS OF LOCATION 6
 24 000016  012737  000340  000006              MOV     #340,@#6        ;VECTOR PSW TO DISABLE INTERRUPTS
 25 000024  010706                              MOV     PC,SP           ;TEMP STORAGE FOR PC
 26 000026  062706  000010                      ADD     #10$-.,SP       ;MAKE RELATIVE ADDR ABSOLUTE
 27 000032  010637  000004                      MOV     SP,@#4          ;SET VECTOR PC FOR TIMEOUT TRAP
 28 000036  012706  000014              10$:    MOV     #14,SP          ;RESTORE STACK POINTER
 29 000042  005045                              CLR     -(R5)           ;TIMEOUT OR DO A DUMMY WRITE
 30 000044  010506                              MOV     R5,SP           ;FALL THRU MEANS GOOD ADDR. PUT IT IN SP
 31 000046  010037  000004                      MOV     R0,@#4          ;RESTORE CONTENTS OF LOCATION 4
 32 000052  010137  000006                      MOV     R1,@#6          ;RESTORE CONTENTS OF LOCATION 6
 33                                     ;
 34                                     ;
```

```
                                        .SBTTL PROMPT
                                ;
                                ;
                                ;       ******************************************************
                                ;       *                                                    *
                                ;       *       PROMPT ROUTINE:                              *
                                ;       *                                                    *
                                ;       ******************************************************
                                ;
000056  000005                  PROMPT: RESET                           ;CLEAR DEVICE REGISTERS OF JUNK CHARS
000060  012703  000010                  MOV     #PRSIZ,R3               ;NO OF CHARS TO PRINT
000064  010702                          MOV     PC,R2                   ;GET RELATIVE OFFSET
000066  062702  001162                  ADD     #PRMSG-.,R2             ;PRINT BUFFER REL ADDR
000072  004767  000372                  JSR     PC,PRINT                ;PRINT MSG.
```

```
                                               .SBTTL  COMMAND MONITOR ROUTINE
                                ;
                                ;       ************************************************
                                ;       *                                              *
                                ;       *       COMMAND MONITOR:                       *
                                ;       *                                              *
                                ;       ************************************************
                                ;
                                ;
                                ;       CONSOLE TRANSMIT AND RECEIVE BAUD RATES MUST BE EQUAL!!!
                                ;
                                ;       GET COMMAND AND ECHO.  LOW BYTE OF R3 WILL
                                ;       CONTAIN THE COMMAND.
                                ;
                                ;
                                ;
000076  105737  177560          GETCMD: TSTB    @#RCSR          ;KEYBOARD READY?
000102  100375                          BPL     GETCMD          ;NO
000104  113703  177562                  MOVB    @#RBUF,R3       ;YES, PUT CHAR IN R3
000110  042703  177600                  BIC     #177600,R3      ;KEEP ONLY 7 BITS <6:0>
000114  110337  177566                  MOVB    R3,@#XBUF       ;ECHO CHARACTER
000120  022703  000015                  CMP     #15,R3          ;CHECK FOR <CR> (COMMAND TERMINATOR)
000124  001402                          BEQ     TEST            ;YES
000126  010304                          MOV     R3,R4           ;NO.  SAVE POSSIBLE COMMAND IN R4
000130  000762                          BR      GETCMD          ;GET NEXT CHARACTER
000132  112737  000012  177566  TEST:   MOVB    #12,@#XBUF      ;OUTPUT A <LF> CHARACTER
                                ;
                                ;
                                ;       FIND COMMAND FUNCTION.  IF VALID, BRANCH TO ROUTINE.
                                ;       IF NOT VALID, BRANCH TO PROMPT ROUTINE.
                                ;
000140  022704  000122                  CMP     #122,R4         ;READ COMMAND?
000144  001430                          BEQ     LOAD            ;YES
000146  022704  000127                  CMP     #127,R4         ;NO.  WRITE COMMAND?
000152  001556                          BEQ     DUMP            ;YES
000154  022704  000126                  CMP     #126,R4         ;NO.  VERIFY COMMAND?
000160  001404                          BEQ     10$             ;YES
000162  022704  000124                  CMP     #124,R4         ;NO.  TUNE COMMAND?
000166  001403                          BEQ     TUNE            ;YES
000170  000732                          BR      PROMPT          ;NO.  PROMPT
000172  000167  000702          10$:    JMP     VERIFY          ;(TOO FAR TO BRANCH)
```

```
                                                 .SBTTL  TUNE ROUTINE
                                         ;
                                         ;       ************************************************************
                                         ;       *                                                          *
                                         ;       *       TUNE ROUTINE:                                      *
                                         ;       *                                                          *
                                         ;       ************************************************************
                                         ;
                                         ; THE TUNE ROUTINE ALLOWS A CONVENIENT WAY TO OUTPUT A CONTINUOUS
                                         ; STREAM OF ALTERNATE 1 AND 0 BITS FOR TUNING AUDIO CASSETTE HARDWARE.
                                         ; ONCE THE ROUTINE HAS BEEN ENTERED, THE BINARY STREAM STARTS TO THE
                                         ; OUTPUT DEVICE (DEFAULT=THE CONSOLE DISPLAY).  TO CHANGE THE OUTPUT
                                         ; DEVICE, HALT THE PROCESSOR AND PUT THE DESIRED DEVICE CSR IN R1.
                                         ; PRESS P TO CONTINUE.  WHEN DONE, EXIT THE ROUTINE BY PRESSING ANY
                                         ; CONSOLE KEY.
                                         ;
000176  012701  177564                   TUNE:   MOV     #177564,R1      ;DEFAULT OUTPUT DEVICE CSR
000202  105711                           10$:    TSTB    (R1)            ;READY FOR CHARACTER?
000204  100376                                   BPL     10$             ;NO, WAIT
000206  112761  000125  000002                   MOVB    #125,2(R1)      ;YES, OUTPUT ALTERNATE 1S AND 0S
000214  105737  177560                           TSTB    @#RCSR          ;OPERATOR WANT SOMETHING?
000220  100716                                   BMI     PROMPT          ;YES, EXIT TUNE ROUTINE
000222  000767                                   BR      10$             ;NO, OUTPUT ANOTHER BYTE
                                         ;
000224  000714                                   BR      PROMPT          ;RESTART
                                         ;
```

```
                                        .SBTTL  READ (LOADER) ROUTINE
                                ;
                                ;
                                ;       ****************************************************
                                ;       *       READ ROUTINE:                              *
                                ;       *                                                  *
                                ;       *       THIS IS AN ABSOLUTE/RELOCATABLE LOADER.    *
                                ;       *       REGISTER FUNCTIONS ARE AS FOLLOWS:         *
                                ;       *                                                  *
                                ;       *       R0=CHECKSUM             R4=WORD REGISTER   *
                                ;       *       R1=LOAD ADDRESS         R5=INPUT DEVICE CSR *
                                ;       *       R2=BYTE COUNT           R6=STACK POINTER   *
                                ;       *       R3=BYTE REGISTER                           *
                                ;       *                                                  *
                                ;       *       A GOOD LOAD RESULTS IN AUTOSTART OR PROMPT. *
                                ;       *       AN ERROR RESULTS IN A MESSAGE, HALT, AND   *
                                ;       *       JUMP FOR RELOAD.                           *
                                ;       *                                                  *
                                ;       *                                                  *
                                ;       ****************************************************
                                ;
                                ;
 000226  012703  000165         LOAD:   MOV     #LDSIZ,R3       ;LOAD MSG SIZE
 000232  010702                         MOV     PC,R2           ;GET BASE ADDRESS
 000234  062702  001346                 ADD     #LDMSG-.,R2     ;ADD OFFSET TO MESSAGE TABLE
 000240  004767  000224                 JSR     PC,PRINT        ;GO PRINT THE MESSAGE
 000244  012705  177560                 MOV     #RCSR,R5        ;DEFAULT READ DEVICE
 000250  005001                         CLR     R1              ;CLEAR THE LOAD ADDRESS
                                                                ;DEFAULT = 0 (NO RELOCATION)
 000252  000000                         HALT                    ;ALLOW OPERATOR TO SPECIFY LOAD
                                                                ;RELOCATION BIAS AND READER DEVICE
                                ;
 000254  042701  000001                 BIC     #1,R1           ;FORCE START ON WORD BOUNDARY
 000260  010146                         MOV     R1,-(SP)        ;SAVE RELOCATION VALUE ON STACK
 000262  005000                         CLR     R0              ;INITIALIZE CHECKSUM
                                ;
                                ; LOOK FOR THE BEGINNING OF A BLOCK ('1' FOLLOWED BY '0' BYTES)
                                ;
 000264  011601                 LDRLUP: MOV     (SP),R1         ;RESTORE RELOCATION BIAS
 000266  004767  000062                 JSR     PC,READIN       ;READ A BYTE
 000272  105303                         DECB    R3              ;CHECK FOR '1' BYTE
 000274  001373                         BNE     LDRLUP          ;LOOP UNTIL BYTE IS FOUND
 000276  004767  000052                 JSR     PC,READIN       ;READ ANOTHER BYTE
 000302  105703                         TSTB    R3              ;CHECK IF '0' BYTE
 000304  001367                         BNE     LDRLUP          ;CONTINUE IF 0; OTHERWISE RESTART
                                ;
                                ; INPUT AND SAVE BYTE COUNT.  IF BYTE COUNT=6, IT IS AN END BLOCK
                                ;
 000306  004767  000066                 JSR     PC,GETWRD       ;GET FULL BYTE COUNT FOR THIS BLOCK
 000312  010402                         MOV     R4,R2
 000314  062702  177774                 ADD     #-4,R2          ;SUBTRACT 4 TO MAKE BYTE COUNT CORRECT
 000320  022702  000002                 CMP     #2,R2           ;WAS BYTE COUNT=6?
 000324  001435                         BEQ     LENDBL          ;YES, IT IS AN END BLOCK
 000326  004767  000046                 JSR     PC,GETWRD       ;NO, GET LOAD ADDRESS
 000332  060401                         ADD     R4,R1           ;SAVE IT IN R1 AS LOADER POINTER
```

```
                                ;
                                ; READ IN REMAINDER OF DATA.  IF THE LOADER HALTS, A CHECKSUM ERROR HAS
                                ; OCCURED; R3=EXPECTED CHECKSUM; R0=CHECKSUM DIFFERENCE ERROR VALUE.
                                ;
000334  004767  000014          30$:    JSR     PC,READIN       ;READ A BYTE
000340  002003                          BGE     45$             ;BRANCH IF MORE DATA REMAINS
000342  105700                          TSTB    R0              ;IF CHECKSUM IS:
000344  001747                          BEQ     LDRLUP          ;OK - CONTINUE
000346  000437                          BR      CKSUM           ;BAD CHECKSUM - INFORM OPERATOR AND HALT
000350  110321                  45$:    MOVB    R3,(R1)+        ;STORE THE BYTE
000352  000770                          BR      30$             ;LOOP FOR NEXT BYTE
                                ;
                                ; INPUT A BYTE, DECREMENT BYTE COUNT, AND ACCUMULATE CHECKSUM
                                ;
000354  105215                  READIN: INCB    (R5)            ;START READER
000356  105715                  60$:    TSTB    (R5)            ;READY?
000360  100376                          BPL     60$             ;NO
000362  116503  000002                  MOVB    2(R5),R3        ;GET CHARACTER
000366  042703  177400                  BIC     #177400,R3      ;MASK OFF HIGH BYTE
000372  060300                          ADD     R3,R0           ;ADD TO CHECKSUM
000374  005302                          DEC     R2              ;DECREMENT BYTE COUNT
000376  000207                          RTS     PC
                                ;
                                ; INPUT AND ASSEMBLE ONE FULL WORD OF DATA
                                ;
000400  004767  177750          GETWRD: JSR     PC,READIN       ;GET ONE CHARACTER
000404  010304                          MOV     R3,R4           ;SAVE IN R4 AS LOW BYTE
000406  004767  177742                  JSR     PC,READIN       ;GET HIGH BYTE
000412  000303                          SWAB    R3              ;PLACE IN HIGH BYTE
000414  050304                          BIS     R3,R4           ;ASSEMBLE BOTH BYTES INTO ONE WORD
000416  000207                          RTS     PC              ;RETURN
                                ;
                                ; CHECK FOR VALID JUMP ADDRESS FOR AUTOSTART
                                ;
000420  004767  177754          LENDBL: JSR     PC,GETWRD       ;GET TRANSFER ADDRESS
000424  004767  177724                  JSR     PC,READIN       ;GET CHECKSUM
000430  105700                          TSTB    R0              ;IF INCORRECT, HALT TO
000432  001005                          BNE     CKSUM           ;INFORM OPERATOR
000434  006204                          ASR     R4              ;GET LOW ORDER BIT
000436  103001                          BCC     AUTOST          ;AUTOSTART IF ADDRESS IS EVEN
000440  000606                          BR      PROMPT          ;INFORM OPERATOR OF GOOD LOAD VIA PROMPT
000442  006304                  AUTOST: ASL     R4              ;RESTORE REGISTER TO ABSOLUTE ADDRESS
000444  000114                          JMP     (R4)            ;GO TO AUTOSTART ADDRESS
```

```
                                      .SBTTL  PRINT SUBROUTINE
                              ;
                              ;
                              ;       ***********************************************
                              ;       *                                             *
                              ;       *       'LOAD ERROR' MESSAGE ROUTINE:         *
                              ;       *                                             *
                              ;       ***********************************************
                              ;
                              ;
 000446  012703  000010       CXSUM:  MOV     #BADSZ,R3
 000452  010702                       MOV     PC,R2
 000454  062702  000604               ADD     #BADMS-.,R2
 000460  004767  000004               JSR     PC,PRINT
 000464  000000                       HALT                    ;CONTINUE?
 000466  000676                       BR      LDRLUP          ;YES, TRY AGAIN
                              ;
                              ;
                              ;
                              ;       ***********************************************
                              ;       *                                             *
                              ;       *       PRINT SUBROUTINE FOR ALL OPERATOR MESSAGES: *
                              ;       *                                             *
                              ;       ***********************************************
                              ;
                              ;               R1=CALLER'S RETURN ADDRESS
                              ;               R2=CURRENT PRINT BUFFER ADDRESS
                              ;               R3=CURRENT NUMBER OF CHARACTERS IN BUFFER
                              ;               XCSR=177564
                              ;               XBUF=177566
                              ;
 000470  105737  177564       PRINT:  TSTB    @#XCSR          ;OUTPUT DEVICE READY?
 000474  100375                       BPL     PRINT           ;NO, WAIT UNTIL READY
 000476  112237  177566               MOVB    (R2)+,@#XBUF    ;YES, PRINT CHARACTER
 000502  005303                       DEC     R3              ;R3=NO. CHARS REMAINING
 000504  001371                       BNE     PRINT           ;IF NE, CHARS REMAIN
 000506  000207                       RTS     PC              ;NONE REMAIN, RETURN
                                                              ;TO CALLER
                              ;
```

```
                                 .SBTTL  WRITE ROUTINE
                        ;
                        ;
                        ;        ***********************************************
                        ;        *                                             *
                        ;        *      WRITE ROUTINE:                         *
                        ;        *                                             *
                        ;        ***********************************************
                        ;
                        ;        WRITE MESSAGE:
                        ;
 000510  012703  000312 DUMP:    MOV     #DMPSIZ,R3      ;PASS NUMBER OF CHARS VIA R3
 000514  010702                  MOV     PC,R2
 000516  062702  000552          ADD     #DMPMSG-.,R2    ;PASS BUFFER ADDRESS VIA R2
 000522  004767  177742          JSR     PC,PRINT        ;GO PRINT MSG.
                        ;
                        ;
                        ; GET READY FOR OPERATOR INPUTS
                        ;
 000526  005000                  CLR     R0              ;CLEAN CHECKSUM
 000530  005001                  CLR     R1              ;DEFAULT LOW ADDRESS = 0
 000532  005002                  CLR     R2              ;HIGH ADDRESS
 000534  012703  000001          MOV     #1,R3           ;AUTOSTART ADDRESS (DEFAULT=1 HALTS AFTER LOAD)
 000540  012704  177564          MOV     #XCSR,R4        ;DFAULT CSR = CONSOLE XCSR
 000544  012705  001700          MOV     #960.,R5        ;DEFAULT NUMBER OF LEADER FRAMES = 2 SECONDS
                                                         ;AT 4800 BPI.  CHANGE FOR SLOWER BAUD RATES
 000550  000000                  HALT                    ;GET OPERATOR INPUTS
 000552  105727  177562          TSTB    #RBUF           ;PURGE 'P' SO IT WON'T CORRUPT PROMPT
                        ;
                        ; LEADER SUBROUTINE.  R5 CONTAINS NUMBER OF NULL BYTES.
                        ;
 000556  105714         1$:      TSTB    (R4)            ;DEVICE READY?
 000560  100376                  BPL     1$              ;NO, WAIT
 000562  005064  000002          CLR     2(R4)           ;YES, OUTPUT A '0'
 000566  005305                  DEC     R5              ;UPDATE NULL FRAME COUNT
 000570  001372                  BNE     1$              ;CONTINUE UNTIL NULL FRAME COUNT=0
                        ;
```

```
                                    ; OUTPUT MAIN BODY OF DUMP
                                    ;
000572  010205                              MOV     R2,R5           ;R5 NOW BECOMES BYTE COUNTER
000574  160105                              SUB     R1,R5           ;NUMBER OF DATA BYTES TO BE OUTPUT
000576  005205                              INC     R5              ;ADD 1 TO ACCOUNT FOR HIGH BYTE IN LAST WORD
000600  010516                              MOV     R5,(SP)         ;STACK WILL CONTAIN BYTE VALUE
000602  062716  000007                      ADD     #7,(SP)         ;DATA BYTES PLUS FRONT BYTES
                                    ;
000606  105714                      10$:    TSTB    (R4)            ;DEVICE READY?
000610  100376                              BPL     10$             ;NO, WAIT
000612  005264  000002                      INC     2(R4)           ;YES, OUTPUT A '1'
000616  005200                              INC     R0              ;ADD TO CHECKSUM
000620  105714                      20$:    TSTB    (R4)
000622  100376                              BPL     20$
000624  005064  000002                      CLR     2(R4)           ;OUTPUT A '0'
000630  004767  000206                      JSR     PC,OUTWRD       ;OUTPUT BYTE COUNT
000634  010116                              MOV     R1,(SP)         ;PUT ADDRESS IN BYTE BUFFER
000636  004767  000200                      JSR     PC,OUTWRD       ;OUTPUT STARTING ADDRESS
000642  000257                              CCC                     ;CLEAR CONDITION CODE BITS
000644  006005                              ROR     R5              ;MAKE R5 THE WORD COUNTER
000646  012116                      30$:    MOV     (R1)+,(SP)      ;PUT DATA WORD IN BYTE BUFFER
000650  004767  000166                      JSR     PC,OUTWRD       ;OUTPUT DATA WORD
000654  005305                              DEC     R5              ;UPDATE WORD COUNTER
000656  002373                              BGE     30$             ;IF NOT NEG, GET NEXT WORD
                                    ;
000660  005400                              NEG     R0              ;2'S COMPLEMENT OF CHECKSUM
000662  105714                      65$:    TSTB    (R4)
000664  100376                              BPL     65$
000666  110064  000002                      MOVB    R0,2(R4)        ;OUTPUT CHECKSUM FRAME
```

```
                                    ; READY FOR END BLOCK
                                    ;
                                    ;       END BLOCK IS SIX BYTES WITH REAL ADDRESS OUTPUT FROM R3,
                                    ;       OR SIMPLY A VALUE OF '1' TO FORCE ODD ADDRESS; LOADER
                                    ;       WILL EXIT ON COMPLETION IF ADDRESS IS ODD, OR AUTOSTART
                                    ;       AT THE USER-SPECIFIED ADDRESS IF THE ADDRESS IS EVEN.
                                    ;
000672  105714                      ENDBLK: TSTB    (R4)
000674  100376                              BPL     ENDBLK
000676  005064  000002                      CLR     2(R4)           ;OUTPUT '0' FRAME
000702  105714                      10$:    TSTB    (R4)
000704  100376                              BPL     10$
000706  005264  000002                      INC     2(R4)           ;OUTPUT '1' FRAME
000712  105714                      20$:    TSTB    (R4)
000714  100376                              BPL     20$
000716  005064  000002                      CLR     2(R4)           ;OUTPUT '0' FRAME
000722  105714                      30$:    TSTB    (R4)
000724  100376                              BPL     30$
000726  112764  000006  000002              MOVB    #6,2(R4)        ;OUTPUT LOW BYTE OF BYTE COUNT
000734  105714                      41$:    TSTB    (R4)
000736  100376                              BPL     41$
000740  005064  000002                      CLR     2(R4)           ;OUTPUT HIGH BYTE OF BYTE COUNT
000744  105714                      50$:    TSTB    (R4)
000746  100376                              BPL     50$
000750  110364  000002                      MOVB    R3,2(R4)        ;OUTPUT LOW BYTE OF START ADDRESS
000754  010300                              MOV     R3,R0           ;PUT ADDRESS IN CXSUM (LB ONLY IS VALID)
000756  105714                      60$:    TSTB    (R4)
000760  100376                              BPL     60$
000762  000303                              SWAB    R3
000764  060300                              ADD     R3,R0           ;COMPUTE CHECKSUM
000766  110364  000002                      MOVB    R3,2(R4)        ;OUTPUT HIGH BYTE OF START ADDRESS
000772  062700  000007                      ADD     #7,R0           ;ADD '1' FRAME AND BYTE COUNT VALUE
000776  000402                              BR      69$             ;GET OUT OF WAY OF MRV11-AA ENTRY POINT
001000  000167  176774                      JMP     START           ;.=173000 FOR USE IN POWER-UP START
001004  005400                      69$:    NEG     R0              ;GET TWO'S COMPLEMENT
001006  105714                      70$:    TSTB    (R4)
001010  100376                              BPL     70$
001012  110064  000002                      MOVB    R0,2(R4)        ;OUTPUT CHECKSUM
                                    ;
                                    ; OUTPUT 100. BLANK FRAMES
                                    ;
001016  012705  000144                      MOV     #100.,R5
001022  105714                      80$:    TSTB    (R4)            ;DEVICE READY?
001024  100376                              BPL     80$             ;NO, WAIT
001026  005064  000002                      CLR     2(R4)           ;YES, OUTPUT A '0'
001032  005305                              DEC     R5              ;UPDATE NULL FRAME COUNT
001034  001372                              BNE     80$             ;CONTINUE UNTIL NULL FRAME COUNT=0
001036  000167  177014                      JMP     PROMPT          ;DONE, RETURN TO OPERATOR
                                    ;
                                    ; SUBROUTINE TO OUTPUT A WORD FROM THE STACK
                                    ;
001042  004767  000012              OUTWRD: JSR     PC,PUTBYT       ;OUTPUT LOW BYTE
001046  000366  000002                      SWAB    2(SP)           ;PUT HIGH BYTE IN LOW BYTE POSITION
001052  004767  000002                      JSR     PC,PUTBYT       ;OUTPUT HIGH BYTE
001056  000207                              RTS     PC              ;RETURN TO CALLER
                                    ;
                                    ; SUBROUTINE TO OUTPUT A BYTE AND ADD TO CHECKSUM
```

```
 58                                     ;
 59 001060  066600  000004              PUTBYT: ADD     4(SP),R0        ;COMPUTE CHECKSUM
 60 001064  105714                      10$:    TSTB    (R4)
 61 001066  100376                              BPL     10$
 62 001070  016664  000004  000002              MOV     4(SP),2(R4)     ;OUTPUT THE BYTE
 63 001076  000207                              RTS     PC              ;RETURN TO CALLER
 64                                     ;
```

```
                                      .SBTTL  VERIFY ROUTINE
                               ;
                               ;
                               ;      ***********************************************
                               ;      *                                             *
                               ;      *      VERIFY ROUTINE:                        *
                               ;      *                                             *
                               ;      ***********************************************
                               ;
001100  010702                 VERIFY: MOV    PC,R2
001102  062702  000544                 ADD    #VFYMSG-.,R2
001106  012703  000121                 MOV    #VFYSIZ,R3
001112  004767  177352                 JSR    PC,PRINT
001116  012705  177560                 MOV    #RCSR,R5          ;DEFAULT CSR
001122  000000                         HALT                     ;WAIT FOR OPERATOR INPUTS
                               ;
                               ; READ A VALID START OF BLOCK '1,0' SEQUENCE
                               ;
001124  004767  177224         10$:    JSR    PC,READIN         ;READ A BYTE INTO R3
001130  005303                         DEC    R3                ;IF A '1', MAKE IT A '0'
001132  001374                         BNE    10$               ;WAIT FOR A VALID '1'
001134  004767  177214                 JSR    PC,READIN         ;GOT A '1', GET NEXT BYTE
001140  105703                         TSTB   R3                ;IS IT A '0'?
001142  001370                         BNE    10$               ;START OVER IF NOT '0'
                               ;
                               ; GET BYTE COUNT.  PUT IT IN R2.
                               ;
001144  004767  177230                 JSR    PC,GETWRD
001150  010402                         MOV    R4,R2
001152  162702  000004                 SUB    #4,R2
001156  022702  000002                 CMP    #2,R2             ;SEE IF THIS IS AN END BLOCK
001162  001412                         BEQ    VFYGUD            ;BRANCH IF IT IS
                               ;
                               ; GET ADDRESS AND PUT IT IN R1
                               ;
001164  004767  177210                 JSR    PC,GETWRD
001170  010401                         MOV    R4,R1
                               ;
                               ; CHECK ALL DATA BYTES IN BLOCK
                               ;
001172  004767  177156         70$:    JSR    PC,READIN         ;READ BYTE INTO R3
001176  120321                         CMPB   R3,(R1)+          ;COMPARE WITH MEMORY BYTE
001200  001005                         BNE    VBMSG             ;BAD BYTE IF NE
001202  005702                         TST    R2                ;ANY BYTES LEFT?
001204  001372                         BNE    70$               ;BRANCH IF BC NOT 0
001206  000746                         BR     10$               ;GET NEXT (END) BLOCK
                               ;
```

```
                                 ; PREPARE FOR THE GOOD NEWS
                                 ;
 001210  000167  176642          VFYGUD: JMP     PROMPT          ;PROMPT MEANS GOOD TAPE READ
                                 ;
                                 ; PREPARE FOR THE BAD NEWS
                                 ;
 001214  010702                  VBMSG:  MOV     PC,R2
 001216  062702  000042                  ADD     #BADMS-.,R2
 001222  012703  000010                  MOV     #BADSZ,R3
 001226  004767  177236                  JSR     PC,PRINT
 001232  005002                          CLR     R2
 001234  114102                          MOVB    -(R1),R2        ;PUT GOOD BYTE IN R2
 001236  010100                          MOV     R1,R0           ;SAVE GOOD BYTE ADDRESS IN R0
 001240  010301                          MOV     R3,R1           ;PUT BAD BYTE IN R1
 001242  000000                          HALT
 001244  000167  177240                  JMP     DUMP            ;THE LOGICAL PLACE TO GO TO
```

```
  1                                                  .SBTTL  MESSAGE BUFFERS
  2                                         ;
  3                                         ;
  4                                         ;        ************************************************
  5                                         ;        *                                              *
  6                                         ;        *       MESSAGE BUFFERS:                       *
  7                                         ;        *                                              *
  8                                         ;        ************************************************
  9                                         ;
 10                                         ;
 11 001250     012      015      015  PRMSG:  .BYTE   12,15,15,7
    001253     007
 12 001254     101      103      125          .ASCII  /ACU>/
    001257     076
 13           000010                          PRSIZ=.-PRMSG
 14                                   ;
 15 001260     007                    BADMS:  .BYTE   7
 16 001261     052      105      122          .ASCII  /*ERROR*/
    001264     122      117      122
    001267     052
 17           000010                          BADSZ=.-BADMS
 18 001270     012      015      015  DMPMSG: .BYTE   12,15,15,15
    001273     015
 19 001274     052      040      105          .ASCII  /* ENTER VALUES:/
    001277     116      124      105
    001302     122      040      126
    001305     101      114      125
    001310     105      123      072
 20 001313     012      015      015          .BYTE   12,15,15,15
    001316     015
 21 001317     011      122      061          .ASCII  /       R1=FIRST ADDR/
    001322     075      106      111
    001325     122      123      124
    001330     040      101      104
    001333     104      122
 22 001335     012      015      015          .BYTE   12,15,15,15
    001340     015
 23 001341     011      122      062          .ASCII  /       R2=LAST ADDR IN DUMP/
    001344     075      114      101
    001347     123      124      040
    001352     101      104      104
    001355     122      040      111
    001360     116      040      104
    001363     125      115      120
 24 001366     012      015      015          .BYTE   12,15,15,15
    001371     015
 25 001372     011      122      063          .ASCII  /       R3=AUTOSTART ADDR (DFLT=NO AUTOSTART)/
    001375     075      101      125
    001400     124      117      123
    001403     124      101      122
    001406     124      040      101
    001411     104      104      122
    001414     040      050      104
    001417     106      114      124
    001422     075      116      117
    001425     040      101      125
    001430     124      117      123
```

```
     001433    124    101    122
     001436    124    051
  26 001440    012    015    015         .BYTE   12,15,15,15
     001443    015
  27 001444    011    122    064         .ASCII  /       R4=DUMP DEVICE CSR (DFLT=177564)/
     001447    075    104    125
     001452    115    120    040
     001455    104    105    126
     001460    111    103    105
     001463    040    103    123
     001466    122    040    050
     001471    104    106    114
     001474    124    075    061
     001477    067    067    065
     001502    066    064    051
  28 001505    012    015    015         .BYTE   12,15,15,15
     001510    015
  29 001511    011    122    065         .ASCII  /       R5=LENGTH OF LEADER (DFLT=960.)/
     001514    075    114    105
     001517    116    107    124
     001522    110    040    117
     001525    106    040    114
     001530    105    101    104
     001533    105    122    040
     001536    050    104    106
     001541    114    124    075
     001544    071    066    060
     001547    056    051
  30 001551    012    015    015         .BYTE   12,15,15,15
     001554    015
  31 001555    052    040    123         .ASCII  /* START TAPE, PRESS P/
     001560    124    101    122
     001563    124    040    124
     001566    101    120    105
     001571    054    040    120
     001574    122    105    123
     001577    123    040    120
  32         000312                      DMPSIZ=.-DMPMSG
  33 001602    012    015    015  LDMSG: .BYTE   12,15,15,15
     001605    015
  34 001606    052    040    105         .ASCII  /* ENTER LOAD BIAS IN R1 (DFLT=0)/
     001611    116    124    105
     001614    122    040    114
     001617    117    101    104
     001622    040    102    111
     001625    101    123    040
     001630    111    116    040
     001633    122    061    040
     001636    050    104    106
     001641    114    124    075
     001644    060    051
  35 001646    012    015    015  VFYMSG: .BYTE   12,15,15,15
     001651    015
  36 001652    052    040    105         .ASCII  /* ENTER READER CSR IN R5 (DFLT=177560)/
     001655    116    124    105
     001660    122    040    122
     001663    105    101    104
     001666    105    122    040
     001671    103    123    122
     001674    040    111    116
     001677    040    122    065
     001702    040    050    104
     001705    106    114    124
     001710    075    061    067
     001713    067    065    066
     001716    060    051
     001720    012    015    015          .BYTE   12,15,15,15
     001723    015
     001724    052    040    123          .ASCII  /* START TAPE, PRESS P DURING LEADER/
     001727    124    101    122
     001732    124    040    124
     001735    101    120    105
     001740    054    040    120
     001743    122    105    123
     001746    123    040    120
     001751    040    104    125
     001754    122    111    116
     001757    107    040    114
     001762    105    101    104
     001765    105    122
            000121                         VFYSIZ=.-VFYMSG
            000165                         LDSIZ=.-LDMSG;
                                    ;
                                    ;
            000000'                        .END    START
```

20-3

```
-11/AUDIO CASSETTE UTILITY - MACRO V03.01 12-JUL-79 00:25:20 PAGE 20-3
BOL TABLE

DST   000442R        ENDBLK  000672R        LOAD    000226R        RBUF   = 177562        VERIFY  001100R
MS    001260R        GETCMD  000076R        OUTWRD  001042R        RCSR   = 177560        VFYGUD  001210R
SZ = 000010          GETWRD  000400R        PRINT   000470R        READIN  000354R        VFYMSG  001646R
UM    000446R        LDMSG   001602R        PRMSG   001250R        START   000000R        VFYSIZ= 000121
MSG   001270R        LDRLUP  000264R        PROMPT  000056R        TEST    000132R        XBUF   = 177566
SIZ= 000312          LDSIZ = 000165         PRSIZ = 000010         TUNE    000176R        XCSR   = 177564
P     000510R        LENDBL  000420R        PUTBYT  001060R        VBMSG   001214R

BS.  000000     000
     001767     001
ORS DETECTED:  0

TUAL MEMORY USED:  458 WORDS  ( 2 PAGES)
AMIC MEMORY AVAILABLE FOR  43 PAGES
=ACU.MAC

ORS DETECTED:  0
```